Lab Assignment - 03

Total (25marks)

Submission Link: https://forms.gle/Pdn42jpqvgy6YWZa6

Submission Guidelines:

1. For all the problems, you have to take **inputs from a .txt file** , any language is okay, you are allowed to **use built in** list, dictionary, arraylist and queue.
2. For task 4, write the answer in your script, and take a picture.
3. Finally, put all the files in a folder. The folder **must** be named following the format **sec_ID_labNo.** Zip this folder. This will result in a zip format of **sec_ID_labNo.zip**. (Example: 2_20101234_1.zip). Submit this zip in the above-mentioned link.
4. You **MUST** follow all the guidelines, naming/file/zipping convention stated above. ***Failure to do so will result in straight 50%-mark deduction.***

Suppose one fine morning, you woke up in the world of Pokémon. Now you decided to become a Pokémon master just like your childhood idol Ash Ketchum.

In order to do so, you have to defeat the gyms/bad guys in the places you walk into in a limited amount of time. Your goal is to get to the Victory Road as quickly as possible.

Task 1: Graph Building [5 marks]

You were given a map of the region. Create a Graph (preferably with adjacent lists) that represents all the places of the region. To help us calculate, we shall assign a number to each of the places on the map:

Places	**Designated Number**	**Connected Places**
Pallet Town (Starting Point)	1	2
Cerulean City	2	3, 4, 5
Celadon City	3	4,7,11
Lavender Town	4	
Viridian City	5	6, 7
Cinnabar Island	6	7, 8
Fuchsia City	7	11
Safari Zone	8	9, 10
Team Rocket's Lair	9	10
Indigo Plateau	10	11
Pewter City	11	12
Victory Road (Destination)	12	

<u>Sample Inputs:</u> (You can use any of the two samples)

```
12
17
1 2
2 3
2 4
2 5
3 4
3 7
3 11
5 6
5 7
6 7
6 8
7 11
8 9
8 10
9 10
10 11
11 12
```

[Here in the first row, 12 means the number of places. In the second row, 17 means the total number of connections. The third row indicates that 1 and 2 are connected. The same rule applies for the rest of the rows.

```
12
1 2
2 3 4 5
3 4 7 11
4
5 6 7
6 7 8
7 11
8 9 10
9 10
10 11
11 12
12
```

[Here in the first row, 12 means the number of places. In the second row, 1 is connected to 2. In the third row, 2 is connected to 3, 4, and 5. The same rule applies for the rest of the rows. Assume that the first place is the Starting point and the last place is the destination]

Assume that the first place is the Starting point and the last place is the destination]	

Create a graph using the above values!

[Remarks for the STs: No need to evaluate Task 1 as we are not generating any output. If there is a problem in task 1, automatically task 2 and 3 will be affected.

So, if task 2 and 3 outputs match, a student should automatically get marks for task 1, as long as he/she follows the guidelines (E.g. taking input from a file)]

Task 2: Breadth-First Search (BFS) [7.5 marks]

Since you are a genius and you have an idea of the BFS algorithm, you can calculate the least number of cities you need to visit to get to your destination, Victory Road. Implement a BFS algorithm to determine the places you need to visit to get to the victory road!

Sample Pseudocode for the BFS implementation: **(You are allowed to try different approaches with same or less time complexity, but the outputs must match)**

```
visited =[0]*noOfPlacesOrNodes , queue =[]
BFS (visited, graph, node, endPoint)
        Do visited[int(node)-1]     1
        Do queue     append(node)
        While queue not empty
        Do m     pop()
        Print m
        If m= endPoint break
        For each neighbour of m in graph
        If visited[int(neighbour)-1] = 0
                Do visited[int(neighbour)-1]     1
                Do queue     append(neighbour)

#Driver
Read data from input.txt and create a graph
BFS(visited, graph, '1', '12')
```

Sample Inputs:
Same as Task 1

Sample Output:
Places: 1 2 3 4 5 7 11 6 12

Task 3: Depth-First Search (DFS) [7.5 Marks]

Now, imagine your rival, Gary, who was also sent to the Pokémon world, wants to become Pokémon master before you. He is planning to get to Victory Road using the DFS algorithm. Implement a DFS algorithm to determine the places he needs to visit to get to the victory road!

Sample Pseudocode for the DFS implementation: (You are allowed to try different approaches with same or less time complexity, but the outputs must match)

```
visited =[0]*noOfPlacesOrNodes
printed = [] #this will store the graph traversing sequence
DFS_VISIT (graph, node)
        Do visited[int(node)-1]    1
        printed.append(node)
        For each node in in graph[node]
                If node not visited
                        DFS_VISIT (graph, node)

#This part is needed if the graph is disconnected.
DFS (graph, endPoint)
        For each node in in graph
                If node not visited
                        DFS_VISIT (graph, node)
        Print “printed” list in a loop till you get the end point

#Driver
Read data from input.txt and create a graph
DFS(graph, ‘12’)
```

Sample Inputs:
Same as Task 1

Sample Output:
Places: 1 2 3 4 7 11 12

Task 4: Comparison [5 Marks]

Now, **calculate** the time-complexity of the BFS (Task 2) and DFS (Task 3) algorithms you used (both for matrix and adjacency list). Also show who gets to the victory road first and why.

Not mandatory task(But Try it for Yourself): (No marks)

1. Can you print the places' names instead of the designated numbers as output?
2. The given BFS code will not work for disconnected graphs. Can you modify this bfs code so that it can work for disconnected graphs as well?
3. Can you detect the cycle using dfs?

   If you don't know these, try to find it out. Take help from STs and faculties . Remember that: ***a bit more learning will never hurt***.